

cache simulator

Report



spring 19

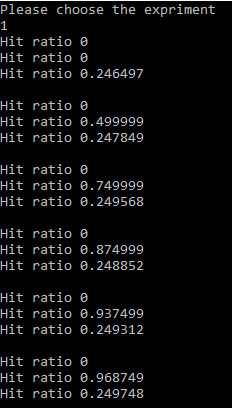
Nada Badawy

Eman Assem

**Project 2: Cache Simulator**

**Experiment 1:**

**results**

In the first experiment we study how the cache performance (hit ratio) changes with the block size, 4, 8, 16, 32, 64, 128 by using the 3 memGen functions.

The first memGen has constant hit ratio which equal to zero regardless the changing in the block size.

For the second memGem, at the first there are direct relation between increasing the size of the block and the hit ratio, till the block reaches to 64 bytes, then the ratio remains constant.

For The third memGem the hit ratio was constant at first till it reaches to 64, then it slightly increased.

**Explanation**

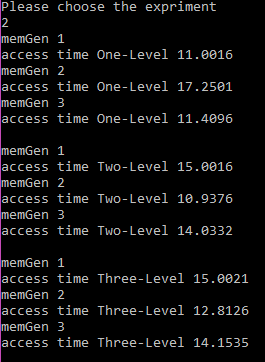
As general, larger blocks should increase the hit ratio, due to special locality. So, we assumed that all the graphs will have a direct relationship between the hit ration and the block size. However, from our result memGen1, memGen2 and memGen3 results in different results and graphs. We expland each reason behind these results.

**First (memGen1):** we have a fixed cache size equal to 64kb, so with block size 4 we have 16384 blocks, with block size 8 we have 8192…etc. The key point is memGen1 generate an address every 16 kb and the block numbers are all multiple of 16, so with direct mapping all the address will map to the same block with all misses which gives 0 hit ratio.

**Second (memGen2):** memgen2 generates address with addition of 4; this feature results in a relation ship between the block size and the hit ratio. However, with the block size equal to 4 the hit ratio is 0, because of with size 4 and the addresses are multiple of 4, then all the addresses mapped to the same block resulting in hit ratio of 0.

**Third (memGen3):** memGen3 generates random addresses between 0 to 256; these results in direct relationship but with small increase compared with memGe2, because in memGen2 we have a sequential test while here we used a random test. For more illustration, the sequential test has more hit ratio than the random one, because the sequential generator of addresses is benefited from the Spatial locality (Items near those accessed recently are likely to be accessed soon).

**Experiment 2:**

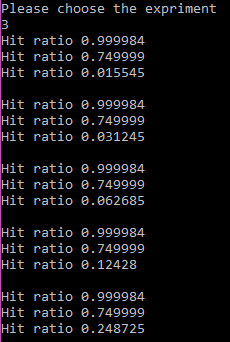
In this experiment we studied how the access time changes with the number of levels of the cache; one level vs two levels vs three levels of cache).

**Explanation:**

for one level it will use the same Tanique for the cache size is equal to 256 KB and the block size equal to 64-bytes. Which means that for memGen1 by generating the addresses by offset 16 will map the addresses among many blocks giving highest hit rate among the three memGens. This results in lowest miss ratio. For the equation AMAT= hit time + Miss rate \* Miss penalty

According to the equation when the miss rate decreases the AMAT also decreases.

**Experiment 3:**

In this experiment we studied how the hit ratio changes with the cache size with a fixed block size: 16 bytes, and cache size varies from 4KB to 64KB in steps that are power of 2, MRU.

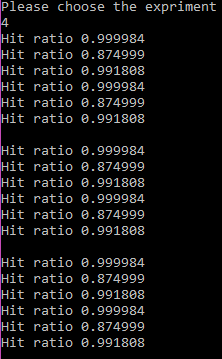
In this experiment memGen1 and memGen2 gives constant hit ratio regardless the changes in the cache size. However, memGen1 gives better hit ratio than memGen2. On the other hand, for the memGen3 the hit ratio increased exponentially.

**Explanation:**

For memGen1 and memGen2, it gives constant hit ratio. The reason behind these results is because of the implementation of the memGens. memGen1 generates an address which is multiple of 16 and the block size itself is 16 bytes which means that for memGen1 it has the highest hit ratio. Because each address will map to a block. The miss will be cold star miss only. Similarly, for memGen2 but the difference is it has less hit ratio because it generates an address with multiple of 4 and the block size is multiple of 16. Thus, for memGen1 and 2 they are independent with cash size.

However, memGen3 shows direct relation between the cache size and hit ratio as assumed before. That’s because the random feature of the address’s generator.

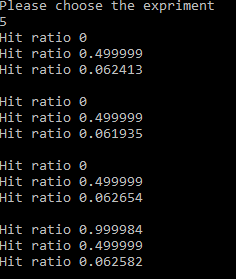
**Experiment4:**

Experiment 4 Studies how the hit ratio changes with the replacement policy with fixed block size equal to 32 bytes and fixed cache size equal to 256KB. The policies are: Random, LRU (Least Recently Used), LFU (Least Frequently Used), and Adaptive LRU/LFU.

For this experiment, changing in the replacement polices does not change the hit ratio for the same memGens, which means that random, LRU and LFU have the same hit ratio for each memgen.

However, memGen1 has the highest hit ratio and memGen2 has the least ratio.

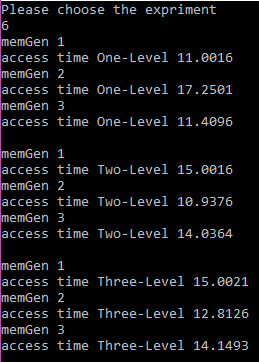
**Experiment 5:**

Experiment 5 study how the hit ratio changes with the number of ways with number of ways equal to 2, 4, 8 and 16 and with constant block size: 8 bytes, and cache size: 16KB, with replacement policy “FIFO (First In First Out)”.

memGen2 and memGen3 gives constant rate ratio regardless the changing in the number of ways. While memGen1 gives 0 till it reaches n=16.

memGen2 gives constant ratio while we assumed it will give direct relation. But the reason behind this may be that memGen2 generates addresses each with multiple of 4 which means each block will be visited two times and that indicates that we will not use the other sets we only use two sets. Moreover, if we compared the results for these experiment with experiment 1 we will find that the hit ratio starts with 0.499999 and then increased but here was constant because here we decreased the block numbers and wasted many sets which is not visited

**Experiment 6:**

Experiment 6 Studied how the access time changes with the number of levels of the cache (one level vs two levels vs three levels of cache). But, with 8-way level cache.

Expermint 6 gives the same results for expermint 2. However, it was 8-way set associative. We can relate experment 5 which indicates that for those memGens the hit ratio did not changed with different number of sets. Thus, with the same tichnique of exprment 2 the results are the same.